Date=14/07/2020
Lecture By=Shubham Joshi
Notes By=Upadhyay Hemanshu
Subject ⇒Python Data Structures

IN PREVIOUS LECTURE **(QUICK RECAP) Date-13/07/2020**	**In today's Lecture (Overview)**
=== We learned More Things About Lists/Types of Lists ⇒ **For Each loop In Python** ⇒ **Enumerate** In Python ⇒ **Comprehension and Slicing** ⇒ **Dictionary In Python** ⇒ **Questions For Self Practice..**	⇒ **Tuples In Python** -What Is mutable -What is immutable ⇒ **Set** In Python ⇒ **Maps** In Python ⇒ **Anagrams** In Python ⇒ **Questions For Self Practice**

⇒ Tuples IN Python

-A **tuple** is a collection of objects which ordered and immutable.

-**Tuples** are **sequences**, just like lists. The differences between **tuples** and lists are, the **tuples** **cannot be changed** unlike lists.

-Tuples are used for grouping data

Example

Create a Tuple:

```
thistuple = ("apple", "banana", "cherry")
print(thistuple)
```

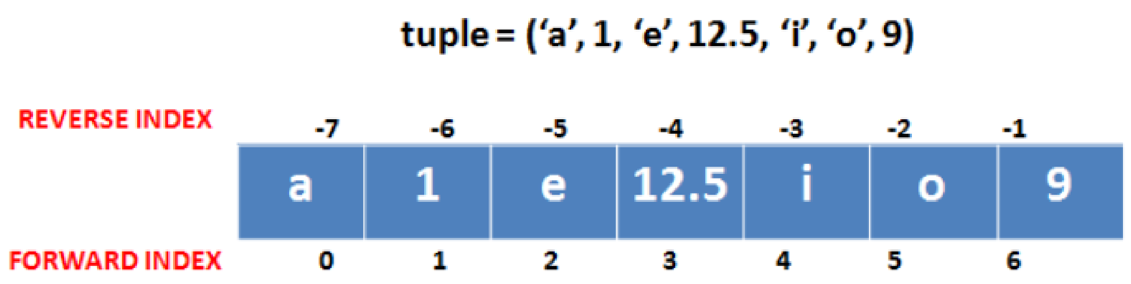
Access Tuple Items

You can access tuple items by referring to the index number, inside square brackets:

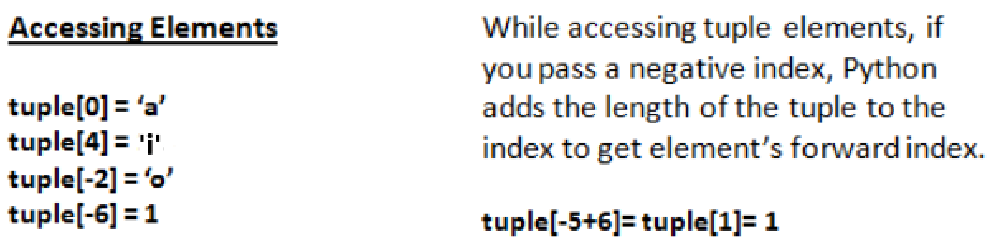
Example

Print the second item in the tuple:

```
thistuple = ("apple", "banana", "cherry")

print(thistuple[1])
```



tuple = ('a', 1, 'e', 12.5, 'i', 'o', 9)



Accessing Elements

tuple[0] = 'a'
tuple[4] = 'i'
tuple[-2] = 'o'
tuple[-6] = 1

While accessing tuple elements, if you pass a negative index, Python adds the length of the tuple to the index to get element's forward index.

tuple[-5+6]= tuple[1]= 1

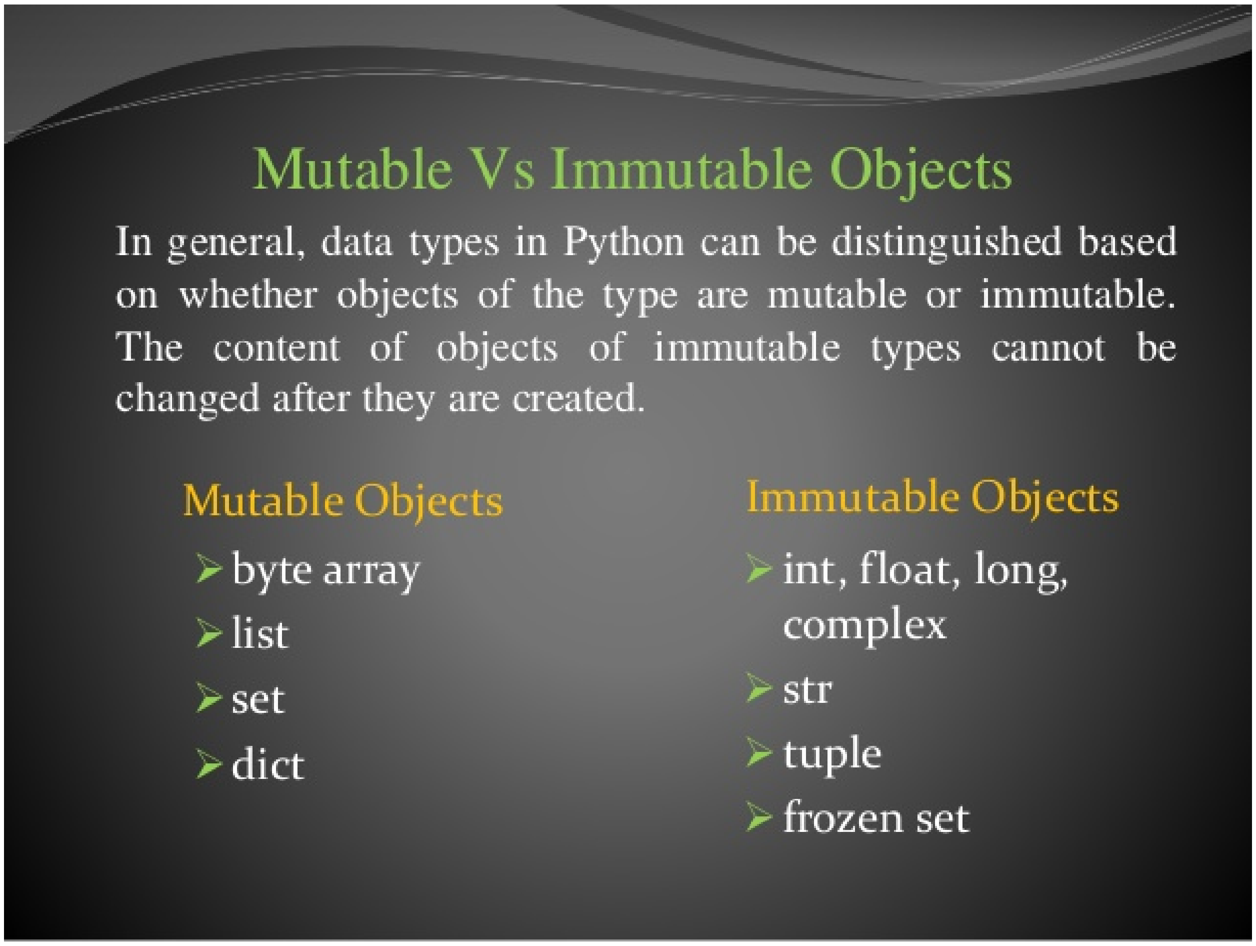
"Click Here" to Know more About It

=What is Mutable In Python??

-Object in python can be changed is called mutable in Python

=**<u>What is Immutable In Python??</u>**

-Object in Python Can’t be changed after it is applied is called immutable in python

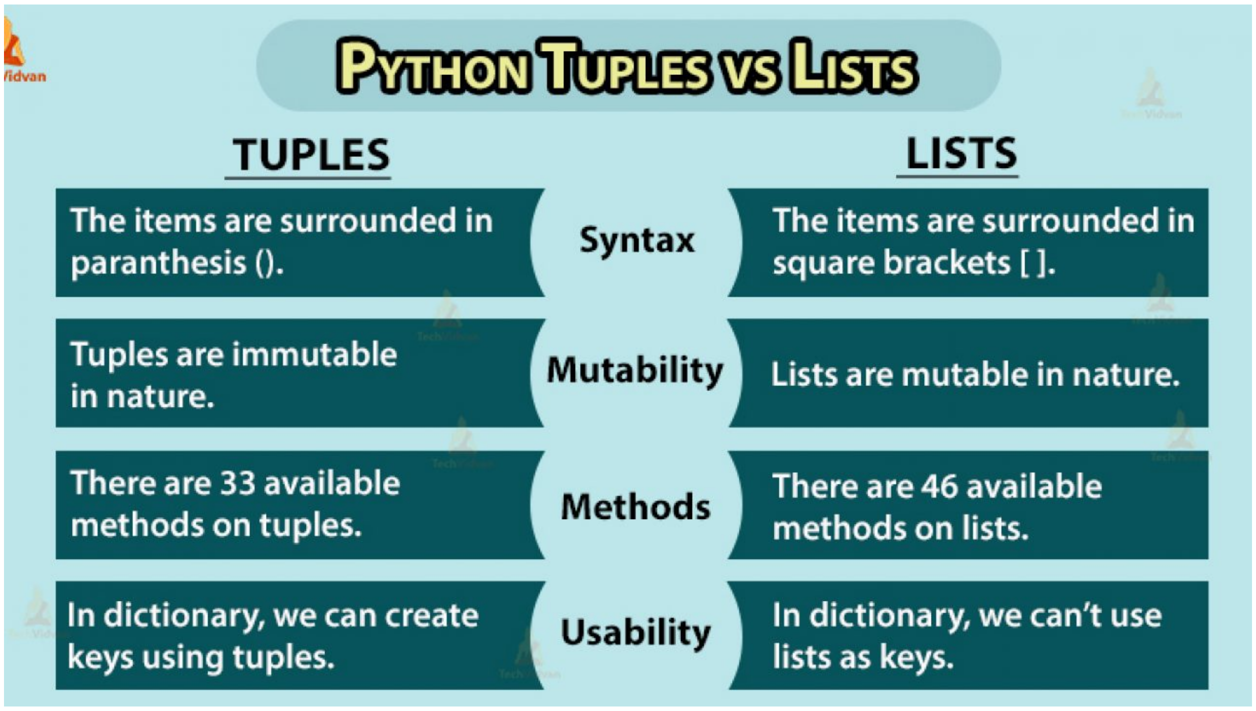


Mutable Vs Immutable Objects

In general, data types in Python can be distinguished based on whether objects of the type are mutable or immutable. The content of objects of immutable types cannot be changed after they are created.

Mutable Objects	Immutable Objects
➢ byte array	➢ int, float, long, complex
➢ list	➢ str
➢ set	➢ tuple
➢ dict	➢ frozen set

Tuples v/s lists



PYTHON TUPLES VS LISTS

TUPLES		LISTS
The items are surrounded in paranthesis ().	Syntax	The items are surrounded in square brackets [ ].
Tuples are immutable in nature.	Mutability	Lists are mutable in nature.
There are 33 available methods on tuples.	Methods	There are 46 available methods on lists.
In dictionary, we can create keys using tuples.	Usability	In dictionary, we can't use lists as keys.

⇒ Set In Python

-A set is a **collection** which is **unordered** and **unindexed**. In Python sets are written with curly brackets.A set is a collection which is unordered and unindexed. In Python sets are written with curly brackets.

Example

Create a Set:

```
thisset = {"apple", "banana", "cherry"}
print(thisset)
```

Note: Sets are unordered In Python so you cannot be sure in which order the items will appear.

Access Items

You cannot access items in a set by referring to an index, since sets are unordered the items has no index.

But you can loop through the set items using a `for` loop, or ask if a specified value is present in a set, by using the `in` keyword.

Example

Loop through the set, and print the values:

```
thisset = {"apple", "banana", "cherry"}

    for x in thisset:

         print(x)
```

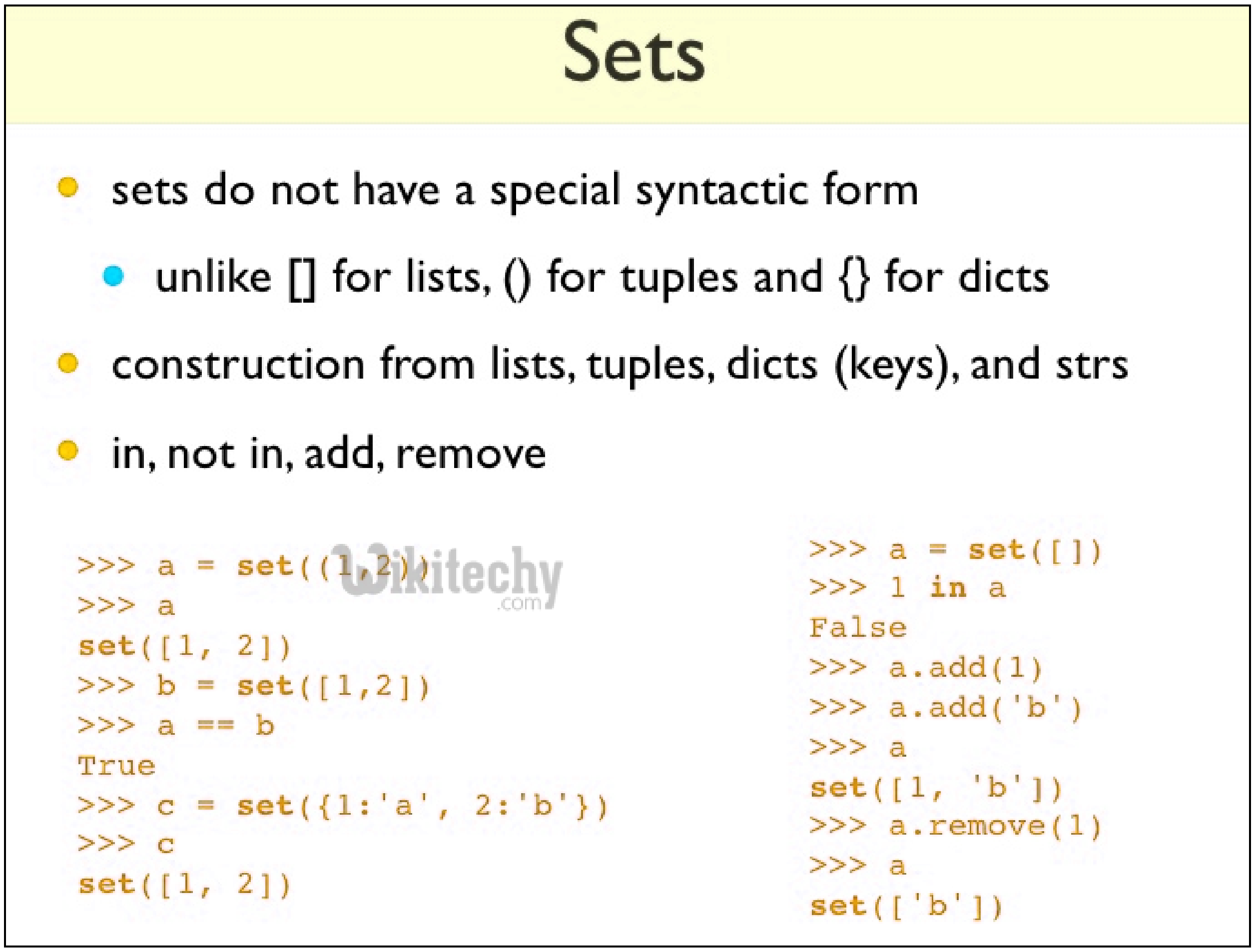
Example

Check if "banana" is present in the set:

```
thisset = {"apple", "banana", "cherry"}

print("banana" in thisset)
```

Change Items

Once a set is created, you cannot change its items, but you can add new items.



Sets

- sets do not have a special syntactic form
  - unlike [] for lists, () for tuples and {} for dicts
- construction from lists, tuples, dicts (keys), and strs
- in, not in, add, remove

```
>>> a = set((1,2))
>>> a
set([1, 2])
>>> b = set([1,2])
>>> a == b
True
>>> c = set({1:'a', 2:'b'})
>>> c
set([1, 2])
```

```
>>> a = set([])
>>> 1 in a
False
>>> a.add(1)
>>> a.add('b')
>>> a
set([1, 'b'])
>>> a.remove(1)
>>> a
set(['b'])
```

"Click Here" to Know more About It

⇒ Maps In Python

-The `map()` function executes a specified function for each item in a iterable. The item is sent to the function as a parameter.

Parameter Values

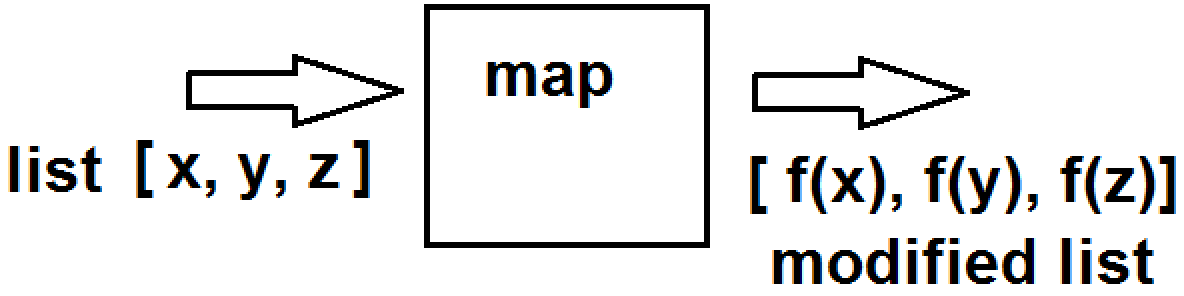
Parameter	Description
function	Required. The function to execute for each item

iterable	Required. A sequence, collection or an iterator object. You can send as many iterables as you like, just make sure the function has one parameter for each iterable.

Example

Make new fruits by sending two iterable objects into the function:

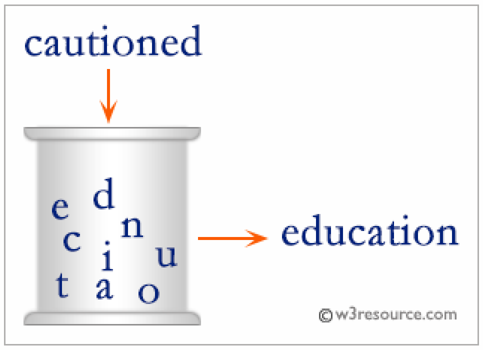
```
def myfunc(a, b):
  return a + b

x = map(myfunc, ('apple', 'banana', 'cherry'), ('orange', 'lemon',
'pineapple'))
```



"Click Here" to Know more About It

⇒ Anagrams In Python

-An anagram of a string is another string that contains the same characters, only the order of characters can be different. For example, “abcd” and “dabc” are anagram of each other.



⇒ Questions For Self Practice

Q1. Write a Python program to create a dictionary from a string. Note: Track the count of the letters from the string. Sample string : 'w3resource' Expected output: {'3': 1, 's': 1, 'r': 2, 'u': 1, 'w': 1, 'c': 1, 'e': 2, 'o': 1}

Q2. https://leetcode.com/problems/valid-anagram/

Q3. Write a Python script to generate and print a dictionary that contains a number (between 1 and n) in the form (x, x*x).